REMARKS

In the forgoing claim amendments, claims 16, 17, 19, 28, 33, 40, 45 and 46 have been amended, and claim 25 canceled. Now pending in the application are claims 16-24 and 26-46, of which claims 16, 33, 45 and 46 are independent.

Claim Amendments

Applicant has amended to clarify the scope of the claimed invention. In particular, claims 16 and 45 have been amended to recite that the user-defined block parameter represents a value to be used during block diagram execution. Claims 16 and 45 have also been amended to recite that the run-time block parameter reduces memory requirements for executing a block diagram. Claims 33 and 46 have been amended to recite that like non-interfaced run-time block parameters are pooled together to reuse data for the like non-interfaced run-time block parameters. Support for the claim amendment can be found in the originally filed specification, claims and figures of the present application. No new matter has been introduced.

Rejection of Claims 16-46 under 35 U.S.C. §112

Claims 16-46 are rejected under 35 U.S.C. §112, second paragraph as being indefinite. The Office Action states that the term "optimally" is subjective and indefinite for failing to provide an objective way to determine an "optimal" run-time block parameter. In response, Applicant has amended the claims to remove the term "optimally". In light of the foregoing claim amendments, Applicant requests that the Examiner reconsider and withdraw the rejection of claims 16-46 under 35 U.S.C. §112, second paragraph, and pass the claims to allowance.

Rejection of Claims 16-46 Under 35 U.S.C. §102

Claims 16-46 are rejected under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,937,257 ("Dunlavey"). Applicant respectfully traverses the rejection for the following reasons.

Summary of the Present Invention

The present invention is directed toward improving a graphical block diagram environment that may be used for modeling systems and analyzing their behavior. System elements are presented as blocks, having inputs and/or outputs and, optionally, parameters that may influence the behavior of the block. One embodiment of the present invention provides a mapping between user-defined block parameters and a run-time version of the parameters to achieve efficient implementation of block equations in the execution (run-time) environment. The parameters may include an interfaced variable and/or a non-interfaced variable. The interfaced variable is a variable whose value can be changed either during simulation or in generated code. In the embodiment of the present invention, the block equations use run-time parameter configurations derived from the parameters entered by the user for an efficient allocation of resources, such as memory space, in the execution of the block equations.

Summary of Dunlavey

Dunlavey is directed to a unit tracking and notification system in a graphical drug model editor. Dunlavey addresses maintaining consistent unit relationships during graphical pharmacological model construction. Dunlavey discloses that the editor enables a user to enter units-specifying data for the objects that represent pharmacokinetic and pharmacodynamic elements. Dunlavey also discloses that when the objects are converted into an internal format, such as a parse tree data structure, the units-specifying data is translated into multidimensional unit type data. Dunlavey further discloses that the multidimensional unit type data is then propagated to identify inconsistent units, and warning messages are displayed when inconsistent units are found. See Dunlavey, column 2, line 62-column 3, line 11.

Claims 16-32 and 45

Independent claims 16 and 45 are directed to a method and a medium, respectively, for processing graphical block parameters in a graphical block diagram environment. The independent claims recite that a user-defined block parameter representing a value to be used during block diagram execution is received and processed to produce a run-time block parameter. The user-defined block parameter reduces memory requirements for executing the block diagram.

Applicant submits that Dunlavey does not disclose *processing a user-defined block parameter to produce a run-time block parameter, wherein the run-time block parameter reduces memory requirement for executing the block diagram*, as recited in claim 16. The Examiner appears to deem that the multidimensional unit type data, which is the internal representation of units-specifying data, disclosed in Dunlavey corresponds to the run-time block parameter of the claimed invention. Applicant respectfully disagrees with the Examiner's position.

An embodiment of the present invention relates to how parameters associated with blocks in a block diagram are processed for executing the block diagram. In the embodiment, a user may specify a block parameter for a block in the block diagram, such as the gain parameter in the gain block (60) depicted in Fig. 3, through a dialog box (70) shown in Fig. 4 or programmatically via a command. The user-defined parameter represents a value to be used during block diagram execution. The user-defined block parameter may be processed by invoking a block method, such as the SetupRunTimeParams() method described in Fig. 11, to produce a run-time block parameter. The user-defined block parameter reduces memory requirements for executing the block diagram. For example, as described in Figure 12 of the pending application, if the value represented by the user-defined parameter exists in the aggregated run-time parameter table, the block's numeric run-time parameter data is freed to reduce memory requirements for executing the block diagram.

In comparison, the multidimensional unit type data disclosed in Dunlavey is used to identify inconsistent units in a graphical drug model. The multidimensional unit type data is propagated to determine whether a user-defined unit is inconsistent with other units in the graphical drug model. The multidimensional unit type data of the Dunlavey reference appears to increase memory requirements for executing the drug model because the multidimensional unit type data includes an array representing a default set of units, such as volume, weight, time, amount and age, and a conversion factor for converting a value entered in one set of units to the default set of units. See Dunlavey, column 3, lines 12-18. The multidimensional unit type data therefore requires more memory space to store a default set of units and a conversion factor. See Dunlavey, column 17, lines 32-46. Dunlavey does not disclose that a run-time block parameter reduces memory requirements for executing a block diagram. The multidimensional unit type data of the Dunlavey reference hence does not provide efficient execution of a block diagram model.

In light of the foregoing reasons, Applicant submits that Dunlavey fails to disclose each and every element of claims 16-32 and 45. Applicant therefore requests the Examiner to reconsider and withdraw the rejection of claims 16-32 and 45 under 35 U.S.C. §102(e), and pass the claims to allowance.

<u>Claims 33-44 and 46</u>

Claim 33 recites a method of mapping graphical block diagram block parameters in a graphical block diagram modeling environment. In the method, a plurality of user-defined block parameters are received and processed to produce a plurality of run-time block parameters. Like non-interfaced run-time block parameters are pooled together to reuse data for the like non-interfaced run-time block parameters. The pooling aspect of the present invention allows the reuse of block parameter data, and hence reduces the memory space allocated for the block parameters in the execution of the block diagram and the generated code for the block diagram. Claim 46 is a medium claim that parallels claim 33.

Applicant submits that Dunlavey does not disclose *pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters*, as recited in claim 33. The Examiner asserts that Dunlavey discloses this feature at column 17, lines 63-65 and Figure 4. See the Office Action, page 4, lines 3-11. Applicant respectfully disagrees.

Dunlavey lists in Fig. 4 the primitives for the internal parse tree data structure for use in translating model blocks into equations. Among the primitives, "SetDiscrete" is used to set a group of categorical variables that are jointly distributed. In Dunlavey, "SetDiscrete" is a primitive used to set a group of categorical variables that are jointly distributed.

at column 17, lines 63-65 and Figure 4. See the Office Action, page 4, lines 3-11. Applicant respectfully disagrees.

Dunlavey lists in Fig. 4 the primitives for the internal parse tree data structure for use in translating model blocks into equations. Among the primitives, "SetDiscrete" is used to set a group of categorical variables that are jointly distributed. In Dunlavey, "SetDiscrete" is a primitive used to set a group of categorical variables that are jointly distributed.

In comparison, the claimed invention pools together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters. For example, an illustrative embodiment may map user-defined block parameters into an existing pool, as recited in claim 34. A value representing the existing pool may also be reused to represent a new user-defined block parameter and therefore save memory space for the new user-defined block parameter. Dunlavey, however, does not disclose that data is reused for the categorical variables in the group.

The Examiner also asserts in the Office Action that Dunlavey discloses at column 19, lines 42-55 that when the code is automatically generated, parameter expressions are maintained in the automatically generated code, as recited in claim 40. See the Office Action, page 6, lines 12-14. Applicant respectfully disagrees. In the claimed invention, the automatically generated code maintains parameter expressions. For example, the parameter expressions may contain non-interfaced variables and/or interfaced variables with mathematical or logical operators. Dunlavey discloses that the code generator (140) and the equation generator (128) translate the internal parse tree data structure into the visible syntax language and Fortran source code. Dunlavey, however, does not disclose that the Fortran source code maintains parameter expressions, as recited in the claimed invention.

In light of the foregoing reasons, Applicant submits that Dunlavey fails to disclose each and every element of claims 33 and 46. Applicant therefore requests the Examiner to reconsider and withdraw the rejection of claims 33-44 and 46 under 35 U.S.C. §102(e), and pass the claims to allowance.

Conclusion

In view of the foregoing, it is respectfully submitted that this application is now in condition for allowance. Should there be any outstanding issues of patentability following the entry of this response, a telephone interview is respectfully requested to resolve such issues.

Respectfully submitted,

LAHIVE & COCKFIELD, LLP

[illegible]

Date: **June 12, 2006**

EuiHoon Lee
Registration No. L0248
Attorney for Applicant
28 State Street
Boston, MA 02109-1784
Tel: (617) 227-7400
Fax: (617) 742-4214